

# 공원 주요시설 및 불법 행위 인식 프로젝트

# Table of Contents

---

01

목 적

02

데이터 전처리

03

학습

04

인퍼런스 결과

---

01

목 적

# 목 적

데이터는 공원 주요시설 및 불법 행위 감시 CCTV 영상 데이터이며, 공원에서 있으면 안 될 것을 CCTV로 감지하는 것이 이 프로젝트의 목적.



전체 데이터 링크

<https://aihub.or.kr/aihubdata/data/view.do?currMenu=115&topMenu=100&dataSetSn=477>

# 목 적

## illegal

쓰레기 봉투, 좌판, 푸드트럭, 현수막,  
텐트, 연기, 불꽃, 반려동물

## legal

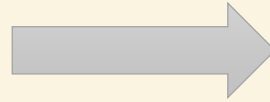
위해방지시설, 벤치, 공원 화분,  
쓰레기통, 휴게공간, 화장실,  
비석, 가로등, 공원 안내표지판

02

데이터 전처리

# 데이터 전처리

JSON에서 YOLO 변환 전		
Train	Images	45,685
	Annotations	46,004
Valid	Images	2,852
	Annotations	2,852
Test	Images	2,846
	Annotations	2,846



JSON에서 YOLO 변환 전후		
Train	Images	45,685
	Labels	43,219
Valid	Images	2,852
	Labels	2,667
Test	Images	2,846
	Labels	2,675


- 원본 Valid에서 62개 이미지 제거, 원본 Train에서는 356개 이미지 제거.
- 잘못 바운딩 박스된 것과 라벨링 된 이미지를 제거 후 데이터를 Train, Valid, Test : 8:1:1 로 분할.
- 위해방지시설과 비석은 인식하기에 모호하여 데이터에 포함하지 않아 Labels의 수가 줄어듬.

```

"images": {
  "ori_file_name": "9_ydsp_su_11-11_10-22-44_for_DF5.jpg",
  "height": 1080,
  "width": 1920,
  "date_captured": "2021/11/11 10:22:44",
  "location": "busan_yongdusan-park",
  "park_type": "neighborhood_park",
  "weather": "sunny"
},
"annotations": [
  {
    "object_id": 9,
    "object_class": "garbage_bag",
    "bbox": [
      [
        849.58984375,
        590.6000000000022
      ],
      [
        1003.5,
        710.1999969482422
      ]
    ]
  }
]

```

- JSON 형태의 라벨 데이터를 YOLO에 맞는 텍스트 형식으로 변환

 15\_etc\_su\_12-09\_09-57-19\_rise\_DF5.txt - Notepad

File Edit Format View Help

```

6 0.0853580729166668 0.11679682131166795 0.05066406250000029 0.0709082765932433
7 0.1009048970540365 0.14354935257523158 0.010481872558593845 0.009058702256944613

```

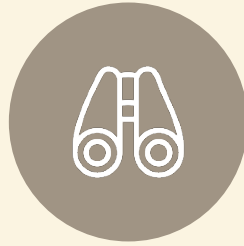


03

학습



Yolov5s

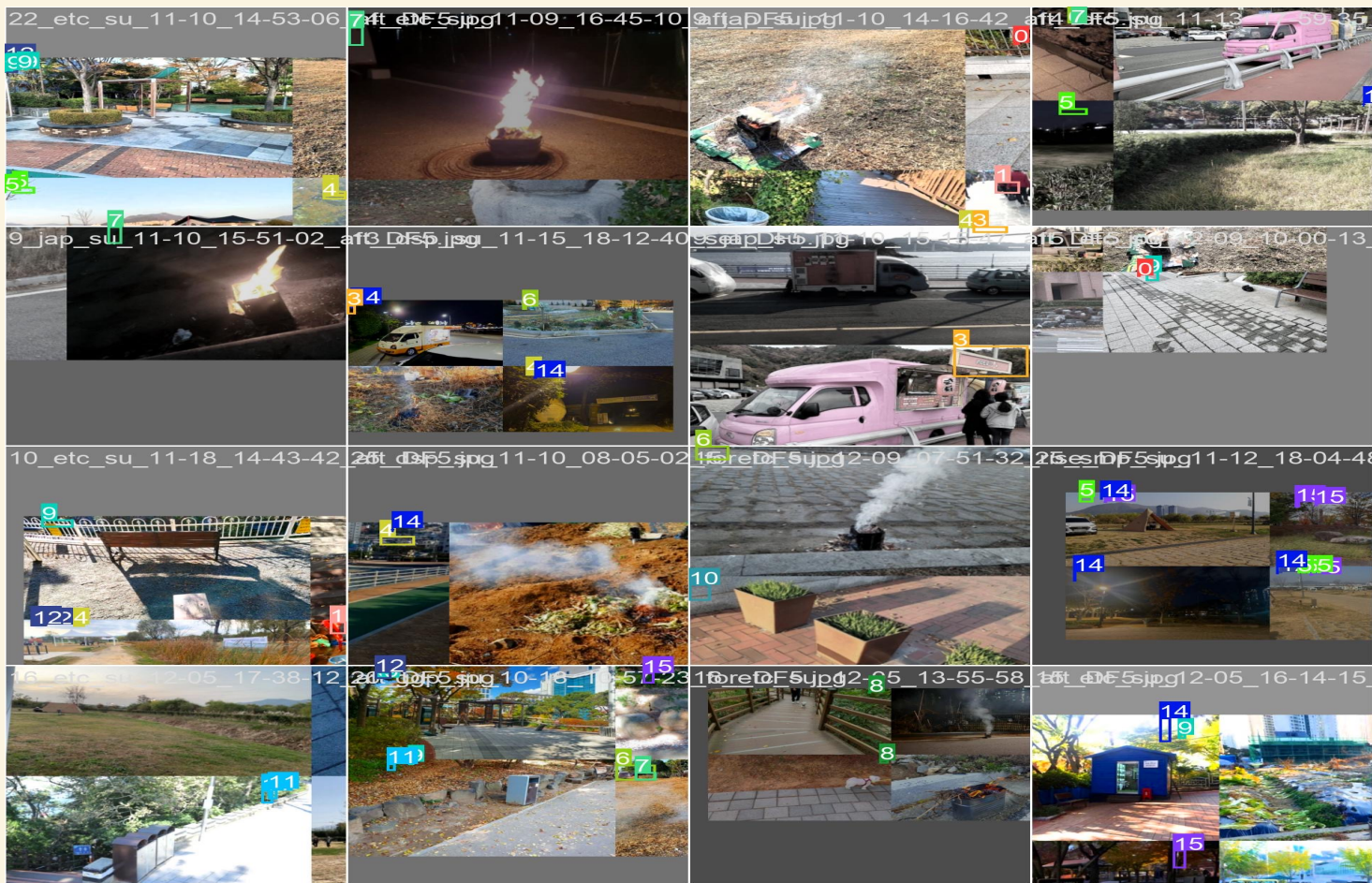


배치 : 80

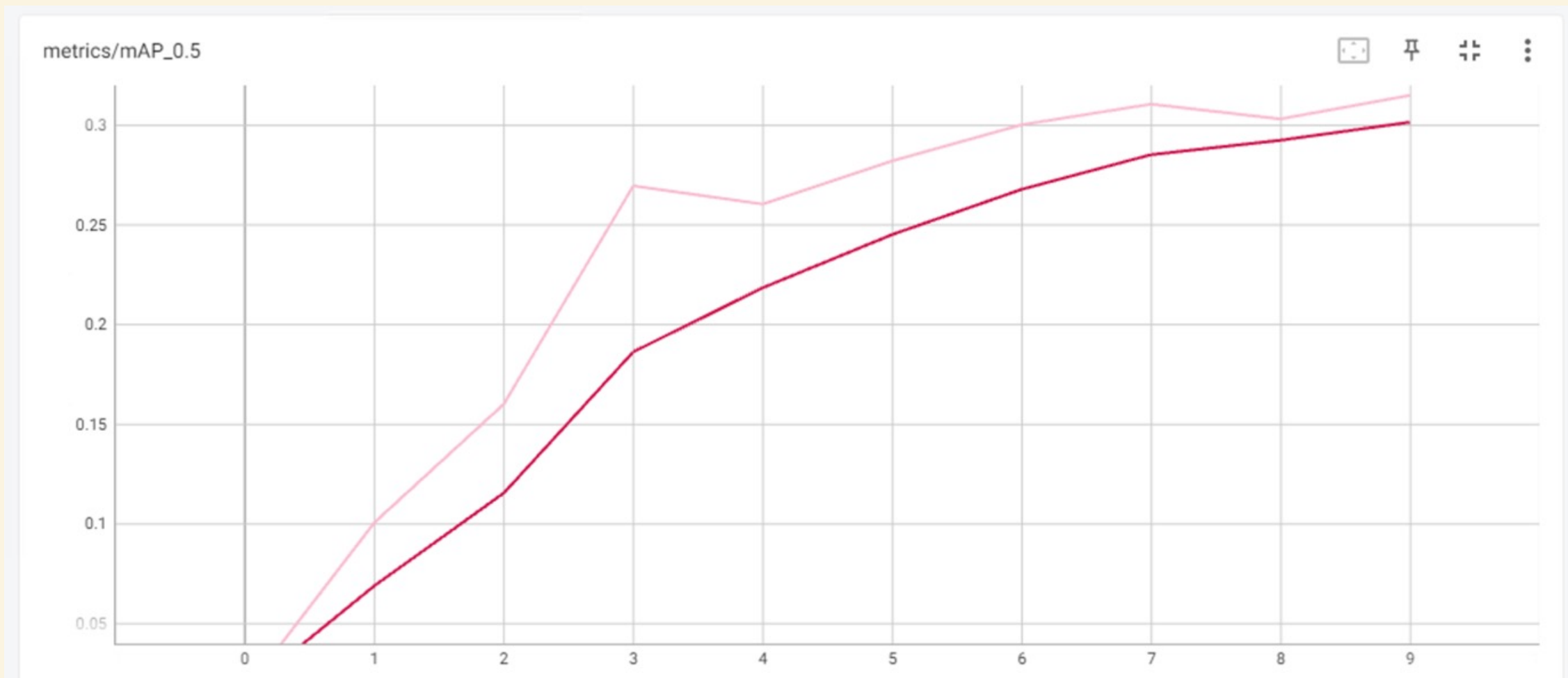


Epoch : 100

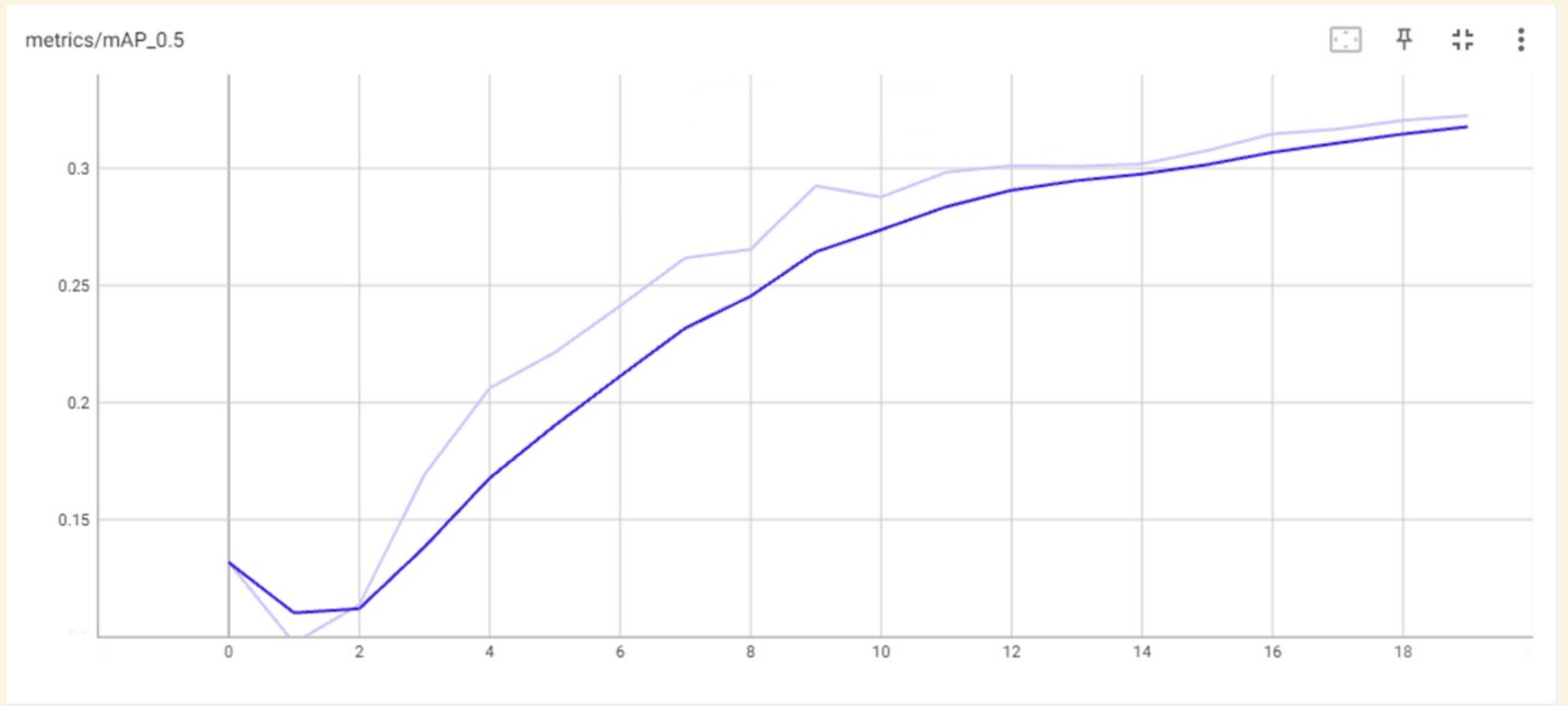
- 비교적 가벼운 5s 모델을 사용하여 배치 사이즈는 Cuda 메모리가 초과되지 않는 80까지 선택 가능하였고, 1 Epoch당 40분 정도 소요되었지만, Epoch에 따른 학습 결과를 확인해 보고자 시간을 들여 100 Epoch로 학습 진행.
- Optimizer 로는 가장 기본이 된다고 생각하는 Adam과 SGD를 사용.



- 학습 진행 후 train batch 확인해본 결과 bbox가 제대로 적용되지 않은 것을 확인함.
- 확인 결과 JSON 라벨 데이터를 YOLO txt 파일로 변환하는 과정에서 실수가 있어 박스의 좌표가 잘못 되어 있는 것을 확인하고 이를 수정하는 작업을 진행.



Adam 을 사용하여 10 epoch 까지 진행된 학습에서 map가 현저히 낮게 나타나는 모습을 보여줌



SGD 또한 마찬가지로 20 Epoch 진행 동안 map\_0.5가 0.3 언저리로 나타나는 모습을 보임

- 수정후에는 정확한 박스 라벨 정보가 들어가서 학습시의 map가 좋게 나오는 모습을 확인

```
Administrator: Command Prompt - nvidia-smi -l
+-----+
| 0 | Tesla V100-PCIE... | TCC | 00000001:00:00:00 Off | 0 |
| N/A | 40C | P0 | 181W / 250W | 15342MiB / 16258MiB | 33% | Default |
+-----+
Processes:
+-----+
| GPU | GI | CI | PID | Type | Process name | GPU Memory |
| ID | ID | ID | | | | Usage |
+-----+
| 0 | N/A | N/A | 7368 | C | ...a3\envs\yolov5\python.exe | 15333MiB |
+-----+
Fri Feb 03 00:17:36 2023
+-----+
| NVIDIA-SMI 456.43 | Driver Version: 456.43 | CUDA Version: 11.1 |
+-----+
GPU Name TCC/WDDM Bus-Id Disp.A Volatile Uncorr. ECC
Fan Temp Perf Pwr:Usage/Cap Memory-Usage GPU-Uutil Compute M.
+-----+
| 0 | Tesla V100-PCIE... | TCC | 00000001:00:00:00 Off | 0 |
| N/A | 39C | P0 | 45W / 250W | 15342MiB / 16258MiB | 0% | Default |
+-----+
Processes:
+-----+
| GPU | GI | CI | PID | Type | Process name | GPU Memory |
| ID | ID | ID | | | | Usage |
+-----+
| 0 | N/A | N/A | 7368 | C | ...a3\envs\yolov5\python.exe | 15333MiB |
+-----+
```

Epoch	GPU_mem	box_loss	obj_loss	cls_loss	Instances	Size
25/99	15G	0.02141	0.01757	0.1279	9	1280: 100% ██████████   2856/2856 [34:52<00:00, 1.36it/s]
	Class	Images	Instances	P	R	mAP50 mAP50-95: 100% ██████████   90/90 [00:39<00:00, 2.29it/s]
	all	2852	5327	0.848	0.846	0.873 0.717
Epoch	GPU_mem	box_loss	obj_loss	cls_loss	Instances	Size
26/99	15G	0.02137	0.01735	0.1279	59	1280: 13% ███   359/2856 [04:22<36:59, 1.13it/s]

# 04

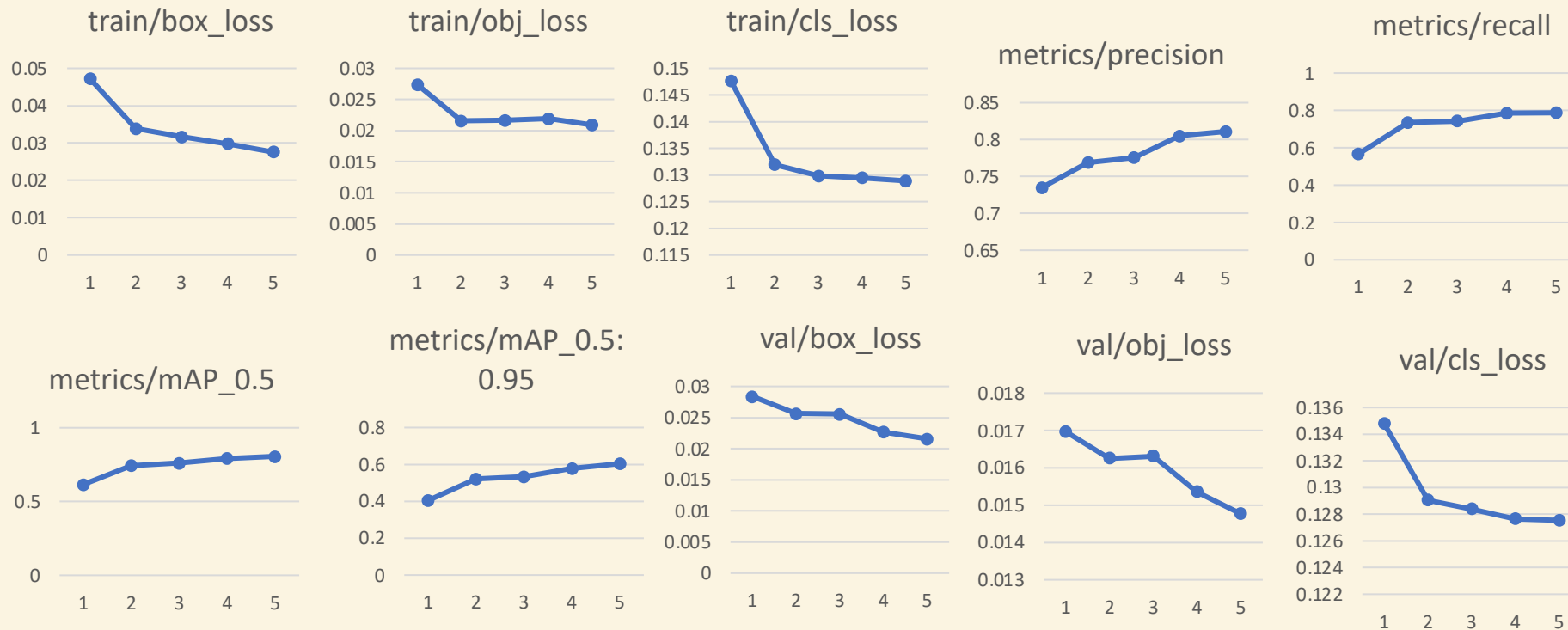
## 인퍼런스 결과

# 인퍼런스 결과

- 편의상 학습에 사용되지 않은 이미지들로 따로 test 데이터셋을 구축 하는 것이 아니라 train val 에 사용된 것과 같은 데이터셋에서 따온 test 데이터셋을 이용하여 inference 코드로 대략적인 모델의 성능을 확인하기로 함.
- 학습에 사용된 데이터와 유사한 이미지 데이터를 사용했다는 점, yolo의 test 코드를 이용한 것이 아니라는 점을 고려하여 결과를 분석해야 할 것으로 생각됨.



# 인퍼런스 결과



# 인퍼런스 결과





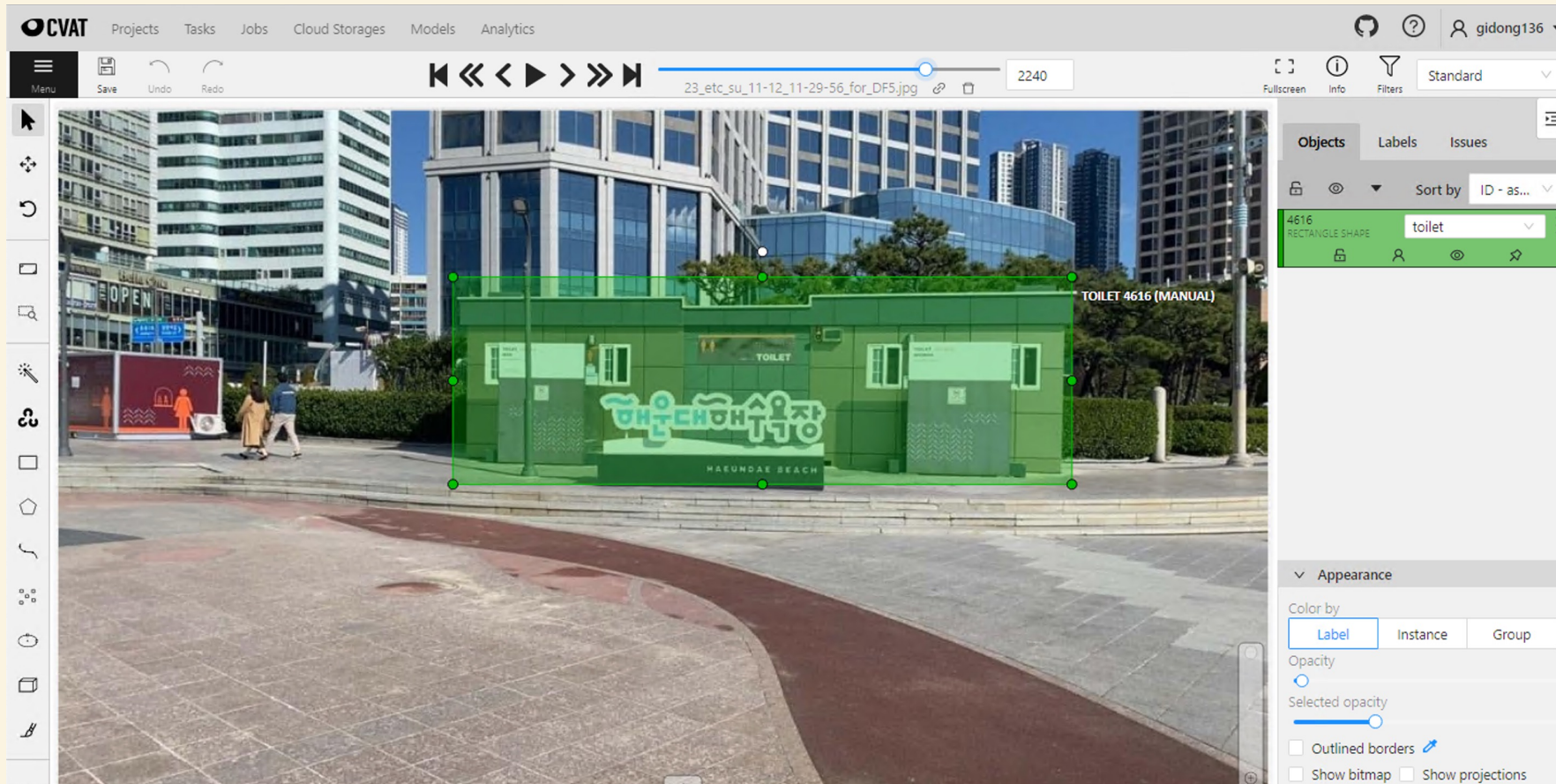
# 인퍼런스 결과



- 종종 쓰레기 봉투와 같은 작은 물체들은 두 개가 있는 상태라도 하나만 인식하는 경우가 발생



# 인퍼런스 결과



- Inference 코드를 통해 탐지한 박스 정보를 파일로 따로 빼내서 CVAT를 통해 해당 이미지에 적용시켜 확인해본 결과

# 프로젝트 총평

- 본격적인 프로젝트가 아니라 제공받은 주제를 가지고 객체 인식 까지만 진행하는 비교적 단순한 프로젝트였던 만큼 YOLO를 이용한 image detection 학습 방법을 익히는 것에 중점을 두고 진행하였습니다.
- AI허브에 있는 데이터셋을 그대로 이용했기 때문에 데이터셋을 구축 하는 데에 큰 어려움은 없었지만 데이터셋의 양이 꽤 많았다는 점과 아무리 이미 완성된 데이터셋이라 하더라도 간간히 있는 휴먼에러를 처리하는 데이터 전처리 과정에 조금 애를 먹었습니다.
- 학습 모델 선정과 Optimizer나 다른 Hyper Parameter 설정에 있어서 근거를 가지고 정한 것이 아닌 편의 위주로 진행되었던 점이 아직은 한계점이라 생각합니다.